



# Enhancing Perception for the Visually Impaired with Deep Learning Techniques and Low-cost Wearable Sensors

Zuria Bauer<sup>a</sup>, Alejandro Dominguez<sup>a</sup>, Edmanuel Cruz<sup>a</sup>, Francisco Gomez-Donoso<sup>a,\*\*</sup>, Sergio Orts-Escolano<sup>a</sup>, Miguel Cazorla<sup>a</sup>

<sup>a</sup>Institute for Computer Research, University of Alicante., P.O. Box 99. 03080, Alicante, Spain.

## ABSTRACT

As estimated by the World Health Organization, there are millions of people who lives with some form of vision impairment. As a consequence, some of them present mobility problems in outdoor environments. With the aim of helping them, we propose in this work a system which is capable of delivering the position of potential obstacles in outdoor scenarios. Our approach is based on non-intrusive wearable devices and focuses also on being low-cost. First, a depth map of the scene is estimated from a color image, which provides 3D information of the environment. Then, an urban object detector is in charge of detecting the semantics of the objects in the scene. Finally, the three-dimensional and semantic data is summarized in a simpler representation of the potential obstacles the users have in front of them. This information is transmitted to the user through spoken or haptic feedback. Our system is able to run at about 3.8fps and achieved a 87.99% mean accuracy in obstacle presence detection. Finally, we deployed our system in a pilot test which involved an actual person with vision impairment, who validated the effectiveness of our proposal for improving its navigation capabilities in outdoors.

© 2021 Elsevier Ltd. All rights reserved.

## 1. Introduction

Vision impairment is a latent problem that currently affects 20 millions of people worldwide. This type of problem does not differ by age or geographical origin. According to the World Health Organization, 253 million people are estimated to live with visual impairment.

General advances in technology, noticeable in the increase of 25 computer computing capabilities and the appearance of wearable devices as well as recent trends, such as deep learning approaches, have contributed to the way in which technological solutions are used and developed to 10 to help the disabled. One of the areas that benefit from this is the learning and recognition 30 of patterns, in which, thanks to these advances, more efficient and effective care frameworks are developed.

In recent years, there has been an increase in the demand for 15 assistive technologies to improve individuals' quality of life. As a result, a number of investigations have emerged. Assistive 35

Computer Vision is one of the growing areas of research. Assistive Computer Vision refers to systems that help people with physical and mental problems to better perform their daily tasks by relying in computer vision techniques such as segmentation or object recognition.

One of the critical environments in which the technology can help the visually impaired is in outdoors. This scenario is filled with potentially dangerous obstacles that could make self-navigation a hard task. For instance, there would be other pedestrians, cars or architectural barriers. The visual impaired are trained to consider the space near to them using the hearing sense and tools like canes or dogs. Nonetheless, the technology could enhance its capacities even more by enlarging the space that they can sense around them and by providing exact descriptions of the environment.

The main goal of this proposal is to develop a system which can sense and understand the space around the user. The proposal is able to provide accurate descriptions of potential obstacles within the environment with the aim of allowing the visual impaired to better navigate in outdoor environments. Our system leverages an ensemble of deep-

<sup>\*\*</sup>Corresponding author  
e-mail: [fgomez@ua.es](mailto:fgomez@ua.es) (Francisco Gomez-Donoso)

**learning techniques to estimate the 3D position of the obstacles and its semantic description. This information is delivered to the user with spoken or haptic feedback.**

Specifically, the main contributions of this work are:

- A wearable system to improve the perception of the visually impaired
- The proposed system is built upon low cost devices
- An urban 2D Object Detector tuned with the auto-updated methodology
- An accurate monocular Depth Estimator for outdoors
- **A pipeline which is finally able to detect the presence of obstacles with a 87.99% accuracy running at 3.8fps.**

The rest of the paper is organized as follows: First, Section 2 presents the state-of-the art in the field. Next, Section 3 describes the proposal in detail. This is followed by Section 4, where the procedures for testing the proposed approach are described and the results of the experiments are presented. Finally, Section 5 includes the discussion and conclusions of the work.

## 2. Related Works

In recent years, many research groups have investigated the development of new sensor-based technologies to aid navigation (improving situational awareness, obstacle avoidance, scene description, etc) for visually impaired individuals. Many of these works have focused on the development of new technologies that enhance/substitute the vision system. In particular, vision substitution systems provide non-display feedback, such as vibration or auditory information. In this category of vision substitution systems for the visually impaired we find three subcategories: Electronic Travel Aid (ETAs), Electronic Orientation Aid (EOAs), and Position Locator Devices (PLDs) (Elmannai and Elleithy, 2017). In the literature, we can find surveys of assistive technologies that review systems within this subcategory (Csapó et al., 2015). Additionally, we can also find recent surveys that analyze research and innovation within the field of mobile assistive technology for the visually impaired (Hakobyan et al., 2013). In this work, we focus on the development of a new system that falls into the Electronic Travel Aid subcategory, so there now follows a brief review of the most recent works on this topic (vision-based).

In Pradeep et al. (2010), a head-mounted, stereo-vision based assistance device is presented that helps the visually impaired to avoid obstacles. There is also an approach that attempts to solve the same problem by using RGB-D cameras (Tian, 2014; Lee and Medioni, 2015). Most of these systems are intrusive and require the individual to wear a bulky camera, which is not practical for everyday use. However, these systems have demonstrated that the visually impaired can benefit from the estimation of depth, helping them to safely navigate the scene (Lee and Medioni, 2016) (avoiding aerial obstacles, stairs, moving objects, etc). These works rely on classic vision algorithms to perform SLAM and to detect obstacles in the surroundings.

With the advent of deep learning techniques and augmented reality headsets, a new trend has recently emerged (Delahoz and Labrador, 2017; Poggi and Mattoccia, 2016; Jafri et al., 2018; Lin et al., 2017). For example, Delahoz and Labrador (2017) presents a deep-learning approach for floor detection. This task is excessively difficult due to the complexity of identifying the patterns of a floor area in many different scenarios. In Poggi and Mattoccia (2016), a wearable mobility aid based on an RGB-D sensor and a deep learning technique is proposed that enables semantic categorization of detected obstacles. However, these systems still require the use of RGB-D sensors and wearable computers to carry out image processing. Lin et al. (2017) proposed a smartphone-based guiding system which uses a real-time object detector Redmon et al. (2016) to detect obstacles in front of the user. Distance to these obstacles is computed based on pixel density, focal distance and camera height. This way of calculating object distance is imprecise and highly dependent on the camera model, sensor noise and other factors, which means the system lacks robustness in estimating precise distances.

An interesting work available in the market is SeeingAI (Microsoft, 2018). This smartphone application uses different deep learning techniques to provide the user with the ability to perform person recognition, text recognition, scene description, etc. However, this application does not allow the user to navigate safely, and does not support external cameras, with the user having to hold the mobile phone while walking.

The approach described in Neto et al. (2017) proposes a face recognition system using a wearable Kinect device. The system is able to recognize a variety of faces and also informs the user of the result. Despite the system seeming to perform accurately, the idea of a wearable Kinect is not appealing. This device is heavy, highly intrusive and requires a power source and a computer for proper use. Finally it is worth noting that the Kinect device performance is impaired in outdoors due to the IR ambient light.

In Wang et al. (2017), a wearable rig with cameras and an embedded computer is proposed to help the visually impaired navigate. Despite the good results, their device is again heavy and highly intrusive. In addition, the three-dimensional camera and the haptic belt are very expensive, which is a severe drawback.

The system introduced in Lakde and Prasad (2015) proposes the utilization of a color camera and an IR sensor to detect obstacles. The proposal distributes the sensors in the users' shoes and a cap. The proposed object detector seems naive as it is based on color information, so its detection power is very limited. In addition, the wiring of all the devices could be uncomfortable and intrusive.

Regarding depth estimation methods, the first work in this field was published in 2005 by A. Saxena Saxena et al. (2006). The paper presented an approach to depth estimation from a single monocular image with a supervised learning approach.

David Eigen published one of the most significant papers in this area (Eigen et al., 2014). The authors used a coarse-scale network to predict the depth of the scene at a global level. This was then refined within local regions by a fine-scale network.

In this way, the local network can edit the global prediction to incorporate finer-scale details.

Two years later, Iro Laina et al. published (Laina et al., 2016). This work was used as the baseline for this part of the present study. They applied a fully convolutional architecture to depth prediction and proposed a more efficient scheme for up-convolutions and combined it with the concept of residual learning for effective upsampling of feature maps. Another innovation was the use of a loss function based on the reverse Huber function (berHu) (Zwald and Lambert-Lacroix, 2012). They achieved the best results in monocular depth to date.

In light of this review of the state of the art, we can conclude that the existing approaches to the problem are somewhat limited. Some of the proposals require either costly or heavy, intrusive devices. Others have technical limitations that render them virtually useless in outdoor environments, or their three-dimensional perception lacks proper accuracy. Nonetheless, the majority of the systems prove that the utilization of speech and haptic feedback helps transmit the information to the user. The works reviewed on monocular depth estimation also highlight the accuracy of these novel systems.

### 3. System Description

As mentioned, we propose a complete framework intended for the visually impaired. The system takes as input the feed from a wearable color camera and provides high-abstraction level descriptions of the potential obstacles in front of the user, so he/she may react accordingly and avoid collisions or dangerous situations. Each description includes distance and relative position of the objects. This information of the scene is summarized and continuously transmitted to the user through the haptic feedback of two smartwatches. In addition, a complete description of the scene can be provided on demand.

#### 3.1. Architecture

In order to capture the environment, the user wears a small size wireless camera, depicted in Figure 1. The smartphone runs a custom application that will receive the images captured by the camera in real time over WiFi. The images are resized to  $608 \times 608$ , which is the input size of the neural network detailed in 3.4. The images are transferred from the smartphone to the remote deep learning server. The resolution shrink is also auspicious since the connection between the two devices is performed over 4G. The images are received by the remote deep learning server, which runs the architectures described in the Subsections 3.3 and 3.4. The images are directly forwarded to the object detection network. The output of this network is the set of detected objects and their corresponding bounding boxes.

The depthmap estimation network is simultaneously executed, but another resize is required in order to feed this network as its input size is  $304 \times 228$ . As a result, this network produces an estimated depthmap of the environment depicted in the input image. The depthmap provides depth values ranging from 0.50 up to 20 meters.

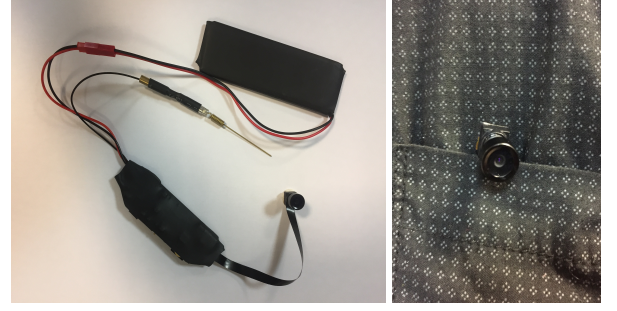


Figure 1: The user wears this small size wireless camera fixed to the pocket of a shirt in order to capture the environment in front of him.

At this point, the system has a depthmap of the scene and the location and category of each object. To estimate the distance to the objects, the following process is carried out: for each bounding box, the points are projected to 3D and the minimum Euclidean distance value inside the bounding box is computed from the estimated depth map. The minimum value is chosen over other statistics as the shape of the objects could be irregular, although the closest point to the user is the minimum distance. In addition, the closest point for each bin is computed in order to notice obstacles that are not detected by the 2D Object Detector. Additionally, the points in the ground plane within a threshold are ignored, otherwise the system would detect the floor as an obstacle.

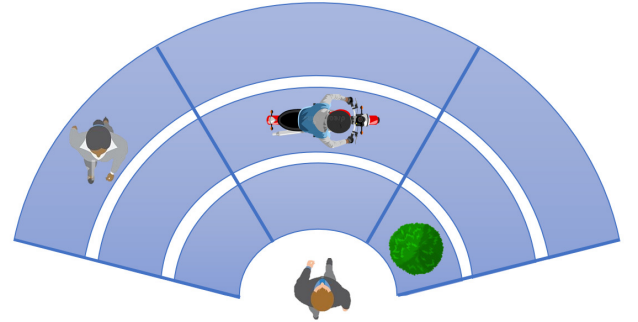


Figure 2: The space in front of the user is discretized, so the obstacles are assigned to a certain bin. In this way, the information is easily converted to haptic feedback to enable fast user awareness.

It is worth noting that the space in front of the user is modeled as a series of truncated semicircles. This representation is discretized in 3 equal range levels, which are also equally divided in 3 sectors each. Figure 2 depicts this discretization of the space. Once the obstacles are detected and their depth position computed, they are assigned to the corresponding bin.

The minimum depth value is constrained by the Depth Estimator, which is 0.50 meters. However, the maximum depth value of this representation is set to 5 meters, as this distance provides enough time to allow the user to properly react to the obstacles. Despite the depth estimation network being capable of predicting values up to 20 meters, points over 5 meters are discarded. This discretization process is carried out in order to simplify the feedback to the user. This representation is then returned to the smartphone.

As already mentioned, the user wears two smartwatches, one on each wrist. Their haptic capabilities are used to continuously transmit the position of the obstacles: haptic feedback on the left or right smartwatch means an obstacle in the leftmost or rightmost bins. If there is an obstacle in the central bins, both smartwatches will show haptic activity.

Three different intensity levels of haptic feedback are used to transmit the depth of the obstacles: stronger feedback means there is an obstacle in the nearest bins while weaker feedback denotes obstacles in the farthest bins.

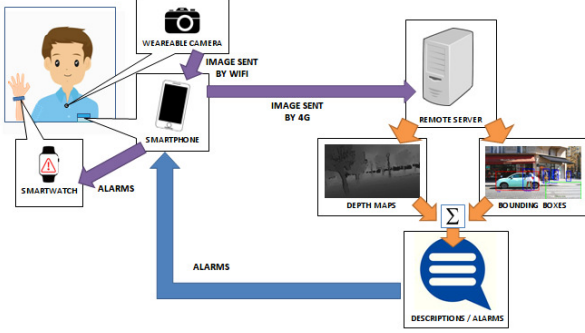


Figure 3: Architecture of the proposed system in this paper.

The haptic feedback is provided in a continuous fashion. Nonetheless, it is possible to obtain a spoken detailed description of the environment on demand by using the smartwatch app. The spoken descriptions consist of short statements describing the position of the detected obstacles that have the following structure: "There is a <object> at <distance> meters in the <nearest|intermediate|farthest> <leftmost|center|rightmost> bin". The tag <object> is filled with the detected object as provided by the object detector, or unknown if the 2D Object Detector does not detect the object. The tag <distance> is filled with the estimated depth that is used to assign the object to a bin, as previously explained. The following tags denote the position of the obstacle in the discretized space. If there are various objects, the descriptions are provided, sorted by distance. For instance, the spoken description of the environment depicted in Figure 2 is: "There is a tree at 0.72 meters on the nearest rightmost bin, a motorbike at 3.14 meters on the intermediate center bin and a person at 4.95 meters on the farthest leftmost bin".

Figure 3 depicts the complete architecture of the system.

### 3.2. Hardware Setup

This section presents the details of the hardware requirements.

The chosen camera is a generic small wireless camera that includes a battery, which is able to keep the camera running up to 2 hours. The battery life-time could be extended by a portable power bank if needed. It provides 25 frames per second and the images are  $1920 \times 1080$  resolution. The camera features a wide angle lens with 140 degrees of angle of view. As mentioned, this camera also includes wireless capabilities, which enables a fluent connection to a smartphone through WiFi.

The smartphone is an Android powered Google Pixel 2 XL, but any mid-range Android smartphone can be used as long as it features WiFi, Bluetooth and 4G capabilities.

Two Microsoft Band are the smartwatches in charge of the haptic feedback. This smartband has a public SDK which allows custom app development on Android hosts.

The most demanding computations are performed on a remote deep learning server. In our implementation, the remote deep learning server featured an Intel i7-7700 CPU with 16 GB DDR4 RAM running Ubuntu 16.04. The server also featured a NVidia Titan X and a NVidia GTX 1080Ti GPUs for deep learning uses.

### 3.3. Depth from Monocular Images Estimation

Depth estimation from monocular frames is a growing field in computer vision. Being able to perform accurate depth map predictions has evident advantages as any color camera would sense three-dimensional data. This capability is important for the present work to be able to create low cost devices for alerting visually impaired individuals of obstacles.

As mentioned, we benchmarked the Depth Estimator from color images approach proposed in Laina et al. (2016) with the proposed dataset. This system describes a fully convolutional neural network that is able to predict depth maps taking a single color image as input. The methodology provided in the original paper was adopted to perform the experimentation.

Architecture 4 features a fully convolutional neural network built upon a ResNet50 (He et al., 2015). This architecture includes different convolution and pooling blocks with eventual residual connections followed by a last fully connected layer first presented as a classifier for the ImageNet challenge. In this incarnation, the last fully connected layer is replaced by a number of Up-Projection layers.

These Up-Projection blocks are presented as the main novelty of this architecture. They are based on the un-pooling method proposed in Dosovitskiy et al. (2014), but extend the idea, introducing residual and projection connections. By chaining up to four of these blocks, this architecture achieves efficient high-level features forwarding while increasing the resolution of the tensors.

Note that the architecture takes  $304 \times 228$  resolution color images as input and predicts  $160 \times 128$  depth maps, which are resized to fit the original image size to allow straightforward alignment with the color image and the 2D object detections. More details of this architecture are provided in Section 4.1.

As mentioned, the network is executed in the remote server, which will also run the 2D Object Detector and the generation of the descriptions and alerts.

### 3.4. 2D Object Detection in Urban Environments

The objective of this system is to create a reliable approach to detect the main objects to be found in any situation in urban environments. For this purpose, we used a state-of-the-art CNN network to detect seven of the most common objects in such



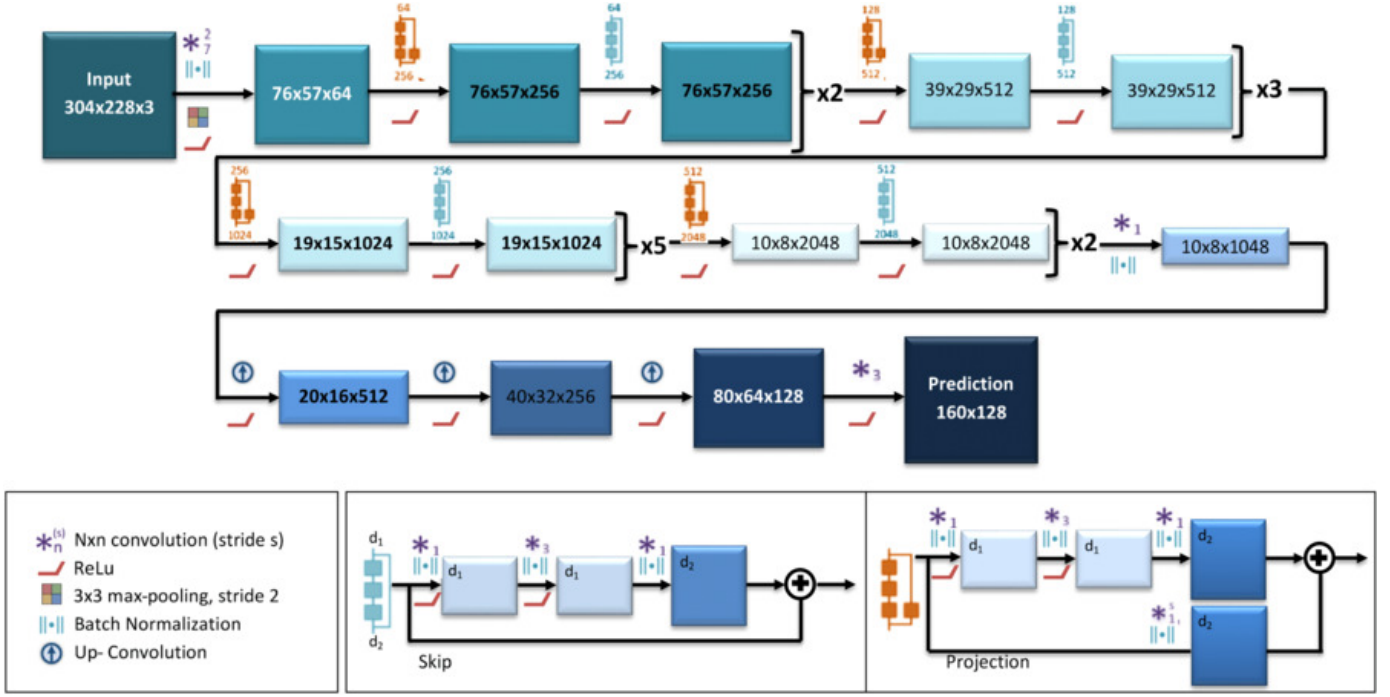


Figure 4: The monocular Depth Estimator is based on a ResNet50. In this case, the last fully connected layer was replaced by several upsampling layers.

environments: people, bicycles, cars, motorbikes, buses, traffic lights and traffic signs.

To achieve this, we rely on a robust detection and classification pipeline. It is necessary not just to be able to classify the main objects but also to find their location within the scene. A region proposal and classification method is used for this task.

We adopted a YOLOv2 (Redmon et al., 2016) for this purpose, modifying it to detect eight classes: seven common traffic items and the background.

The 2D Object Detector training process involved the PASCAL VOC dataset, which is based in Ren et al. (2015). This dataset has various advantages in certain classes. For instance, the category person, included in PASCAL VOC, offers great generalization capabilities. The PASCAL VOC dataset includes many instances of the category people from a variety of angles, sizes and in different light conditions and poses. In an urban scene, 99% of persons are either riding a bicycle, on a motorbike or walking alongside a road. Therefore, we added more of these kinds of objects from our own recordings to the dataset. The same applies to bicycles and motorbikes. Hence, we also added these objects from our real life recordings and from videos from Internet, which we manually annotated.

The UDacity dataset gave us even greater robustness when locating cars, as all the recordings were acquired from a person perspective. UDacity also provides the kind of images and objects that anyone would find in an urban situation: cars and other vehicles from a pedestrian point of view and angle (about 160 cms height). This addition increased the accuracy of our model when detecting cars.

So finally, the ensemble of datasets had around 375,000 annotated objects of which 106,920 images were used for train-

ing. The train/validation/test split sizes are 40% for training, 40% for validation and 20% for testing purposes.

We trained the network for 80,000 epochs, using a learning rate starting at 0.001, and reducing it by a factor of 10 every 20,000 epochs. The momentum was set to 0.9 and weight decay to 0.0005.

This baseline model was then used to perform an auto updated learning, as proposed in Dominguez-Sanchez et al. (2018). This method uses a model trained on one manually annotated dataset to automatically label another dataset. Both datasets (the manually labeled and the automatically labeled) were then used to train a final model. The manually labeled dataset was introduced earlier.

The automatically labeled dataset was composed of around 7h10m of 1280 × 720 video at 60fps in 23 different situations (urban, countryside, roads, etc). We used a camera capable of recording at a good frame rate to get details from any objects (cars, bicycles, etc) moving at high speed. This feature is essential in order to avoid blurry frames caused by low frame rate cameras. The cameras we used for acquiring new data consisted of an HD (1280 × 720 resolution) action camera (H5 Midland). This camera is able to record video at 60fps, and has a CMOS sensor of 5 Mpixels and a wide angle lens of 170°. Moreover, we also used a SONY PlayStation Eye camera, which is able to record videos at 60fps (640 × 480 resolution). It features a VGA CMOS sensor with a wide-angle lens of 75°.

The videos were converted to a lower frame rate in order to avoid recurrent frames. In this case, 10fps provided a reasonable number of frames to train without causing many repetitions as result of the similarity of sequentially recorded frames.

As mentioned earlier, this new collection of samples was annotated by feeding them with the model we trained with the

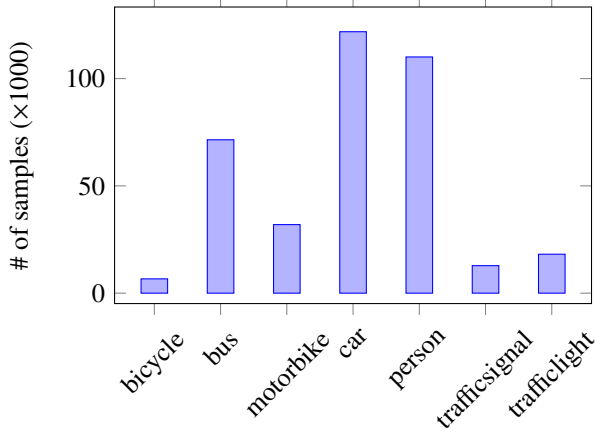


Figure 5: Number of annotations per class after adding automatic annotated images. Note the logarithmic scale in the Y axis.

manually labeled dataset. A final model was then trained on both datasets. The cumulated number of samples per class is depicted in Figure 5. The auto-updated learning process helped to outperform the accuracy of the model trained only with the manually labeled dataset.

### 3.5. Object Tracking

This section introduces the tracking methods in a video feed for improving the performance of the 2D Object Detector. The tracking capabilities helps to reduce the computational load to provide real-time outputs.

The plan to follow in the detection-tracking ensemble was to first detect the objects with the 2D Object Detector, and then track them until they vanished from the images or until the reliability of the tracker decreased sufficiently. This would mean that were no longer confident of the localization of the objects detected in the first place and further detections were needed.

We considered three state-of-the-art tracking algorithms: KCF (Henriques et al., 2014), MedianFlow (Kalal et al., 2010) and Mosse (Bolme et al., 2010).

Tracking objects in a urban environment from the pedestrian point of view has particular drawbacks such as tracking objects near the limits of the camera view, such as parked cars, other pedestrians or moving vehicles. These objects can quickly increment their relative size and location in the scene, and most trackers do not perform well with such large, fast variations. However, objects moving beside or in front of the individual can be tracked for a long time. An example of this can be seen in Figure 6. The experimentation of this system is shown in Section 4.2.

In the context of the proposed system, the tracking is performed in the remote server, which will also run the 2D Object Detector.

## 4. Experiments

In this section, we show the results of the experiments we carried out. First, we put to test the depth estimation from

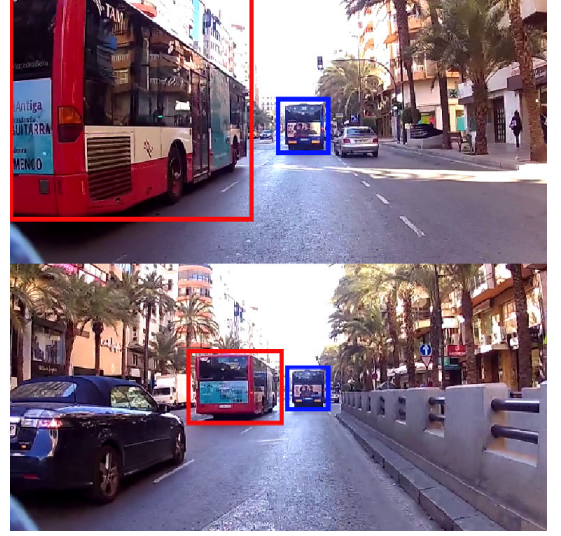


Figure 6: This sequence depicts a bus quickly changing its relative size in a few frames (red) while there is another bus (blue) in the same scene which does not show this behavior due to the distance from the observer.

**monocular frames. We benchmark the model provided by the original authors of the architecture and the model we trained with UASOL, which is our outdoor dataset. Then, we introduce the experimentation for the 2D object detection and tracking in urban environments. In addition, the whole system is tested in sequences of the UASOL dataset. Finally, we describe a pilot test we carried out.**

### 4.1. Depth from Monocular Images Estimation Experimentation

The first experiment is based on the replication of the model proposed by Laina et al. (2016). Following the specifications provided in the paper, the system was trained as detailed: The weights of the ResNet50 were initialized with a pretrained model on the ImageNet dataset, the learning rate was 0.01 and it was gradually reduced every 6-8 epochs, the batch size was 16 and was trained for 20 epochs. The loss function was BerHu (Zwald and Lambert-Lacroix, 2012). Data augmentation was executed as suggested in Eigen (Eigen et al., 2014). Note that the input resolution is  $304 \times 228$  and the estimated depthmaps  $160 \times 128$ . The original model is publicly available but there were several concerns about reproducibility of the model. Thus, we followed the original work to replicate the results. We reported both the root mean square error (RMSE) and the mean absolute relative error (MREl). The MREl is defined as follows:  $\frac{1}{|T|} \sum_{y \in T} |y - y^*|/y^*$ , where  $y^*$  are the predicted values,  $y$  is the corresponding ground truth and  $T$  are the samples.

As shown in Table 1, the results of the two models (original and replicated) did not perform equally. The original model was originally trained using Matconvnet, which appears to be the reason the results obtained by our network differ, ours being trained with Tensorflow.

Despite the reasonably good results obtained using the test set of the NYU dataset, these models did not perform as well outdoors. This is because the NYU dataset is only composed of

Table 1: Comparison of the proposed approach and the original network using the NYU Depth v2 test set.

Architecture and Model	MReIE	RMSE
Laina et al. (2016) (publicly available model)	0.127	0.573
Laina et al. (2016) (trained on the NYU dataset by us)	0.198	0.672

indoor scenes. So decreased accuracy might be expected when tested in outdoor environments.

To make the approach robust outdoors, we relied on the UASOL dataset<sup>1</sup>. UASOL is a Large-scale High-resolution Stereo Outdoor Dataset created at the University of Alicante that features sequences of color images and corresponding depth maps captured in outdoors environments from a pedestrian’s perspective. Different visual features of the environments, weather conditions and moments of the day are considered to provide high variability data. In addition to depth estimation from a single color frame, this dataset could be used for depth estimation from a set of color frames, structure from motion or stereo benchmarking. Thus, we also trained Iro Laina’s network with this dataset.

In order to gain insights into the generalization capabilities of the models, we also tested the models of the network with synthetic data. These experiments helped to calibrate the behavior of the approach in totally different scenes that the network never saw. We extracted about 500 random samples of the UnrealROX dataset (Martinez-Gonzalez et al., 2018) and computed the corresponding error.

The results obtained are shown in Figure 7. As can be seen, the original model released by Iro Laina et al. and the model we trained on the NYU dataset perform similarly. They both predict good outputs for the indoor scenes (NYU and UnrealROX). Neither of the systems is capable of projecting the floor or walls correctly. The original model shows less noise than the model we trained. On the other hand it shows greater problems with shadows than ours. Both models performed poorly in outdoor environments (UASOL).

The results obtained by the network trained with the UASOL dataset show poor performance in indoor scenarios (NYU and UnrealROX). Nonetheless, the predictions for the outdoor samples (UASOL) yield high accuracy. The network is even capable of generating the correct flooring, as well as rendering small details of the scene like posts or streetlights. Furthermore, the error level is quite low. This means that the UASOL dataset provides good quality images and also the adequate number to train the network correctly. The network trained on this dataset is also capable of providing a greater range of depth values (0.5m-20m) than the provided by the model trained on the NYU dataset (0.5m-5m).

Quantitative evaluation of the models and the datasets are provided in Table 2. The conclusions stated before are supported by these results.

Table 2: Comparison of the proposed approach against the original network using the UASOL and UnrealROX test sets.

Test Set	Architecture and Model	MReIE	RMSE
UASOL	Laina et al. (2016) (publicly available model)	0.753	8.119
	Laina et al. (2016) (trained on the NYU dataset by us)	0.327	4.400
	Laina et al. (2016) (trained on the UASOL dataset by us)	0.756	8.1401
UnrealROX	Laina et al. (2016) (publicly available model)	0.998	2.922
	Laina et al. (2016) (trained on the NYU dataset by us)	0.446	1.548
	Laina et al. (2016) (trained on the UASOL dataset by us)	1.535	6.244

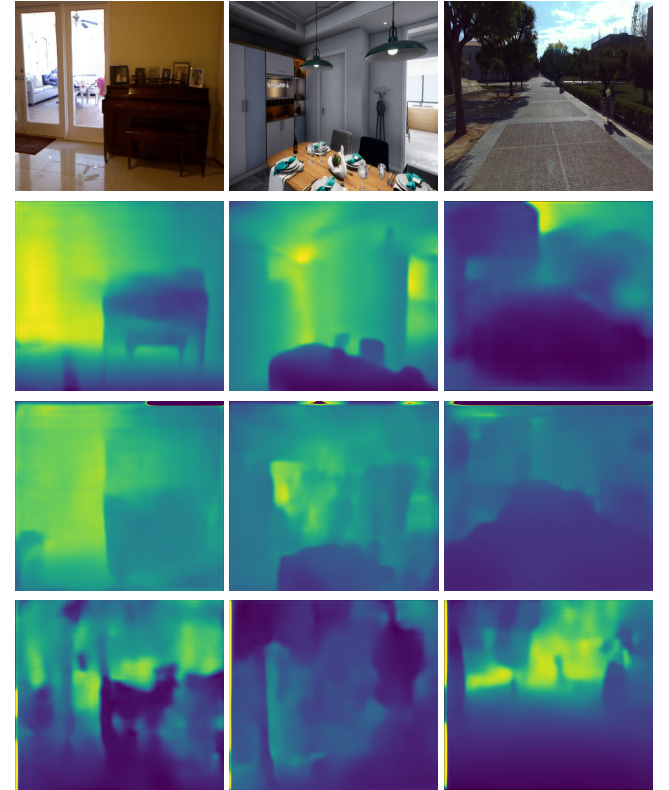


Figure 7: Sample results from each test carried out. The columns contain images for a sample of the NYU dataset, the UnrealROX dataset and the UASOL dataset. The rows are the color images, the predictions provided by Iro Laina’s original model, the predictions provided by Iro Laina’s approach we trained with the NYU dataset, and the predictions provided by Iro Laina’s approach trained with the UASOL dataset.

**In light of the experiments, the architecture is reported to be robust enough to provide accurate depth maps from a single color image. Furthermore, the model we trained with the UASOL dataset performed nicely in outdoor scenarios. As the proposal of this work is intended to be deployed outdoors, this is the model the system finally features.**

Finally, it is worth noting that the experiments were executed on the test bench detailed in Section 3.2. The depth values were reduced by 10% of the ground truth at training time, so the network is forced to learn that the obstacles are slightly nearer than

<sup>1</sup><http://www.rovit.ua.es/dataset/uasol/>



they actually are. This was done to compensate for the latency and the movement of the user. While an image is captured and the output is given, the user might have walked forward, making the distances outdated for that instant. Thus, we deliberately reduced the ground truth depth values to address to his issue.

#### 4.2. 2D Object Detection and Tracking in Urban Environments Experimentation

This section details the experimentation we carried out to validate the 2D Object Detector and the tracker methods focusing on accuracy and inference time. It is worth noting that the experiments were executed on the test bench described in Section 3.2.

The baseline consisted of the 2D Object Detector as described in Section 3.4 with no tracking methods. In this case, we applied a learning rate of 0.0001, and trained for 45,000 iterations, with a momentum of 0.9 and decay of 0.0005. This model achieved 0.62 mAP in the test set.

We noticed that the detection for bicycles, motorbikes and traffic lights was not as accurate as the remaining categories. This is to be expected as the number of bicycle, motorbike and traffic lights samples is lower than the other classes in the manually labeled dataset. Thus, we leveraged the auto-updated learning methodology to automatically annotate new samples. We then retrained the model for 80,000 iterations using this extended dataset. We used a learning rate of 0.001, achieving 0.742 mAP. This increment of the accuracy rate is attained by the substantial improvement in the detection of bicycles, motorbikes and traffic lights, as reported in Table 3. However, the accuracies of all classes improved.

In the proposal that we adopted for the 2D Object Detector (Redmon et al., 2016), the authors set the NMS parameter as 0.7. We tested this parameter by ranging it from 0.6 to 0.9. According to the mAP, the best performer was provided by a NMS threshold of 0.8, which provided a 0.62 and 0.74 mAP as shown in Table 3.

Regarding the tracking methods, we report here the accuracy of the tracking algorithms. This experiment is intended to compare the performance of each tracking method with the 2D object detection executed for each frame. For each frame of a test sequence, the 2D Object Detector was used to predict the bounding boxes of the objects. Each tracking method was then tested by running the detector every 11, 9, 7, 5 and 3 frames. The results of these experiments are shown in Figure 8. As expected, the accuracy decreases as the skipped frames increase. Note also that the best performer in each case is the KCF method. Finally, it can thus be concluded that the best option is to perform a classification every 3 frames and letting the KCF tracker do the rest.

For the runtime test, we set up two scenarios. First, a busy environment, which was taken in a city center with a high density of objects, and a quiet environment which consisted of a sequence recorded in a motorway with a low density. Both sequences were used to test the following setups: the 2D Object Detector executed each frame with no tracking, and KCF, MedianFlow and Mosse tracking methods.

The results are presented in Table 4. As can be seen, MedianFlow and Mosse performed similarly while KCF was

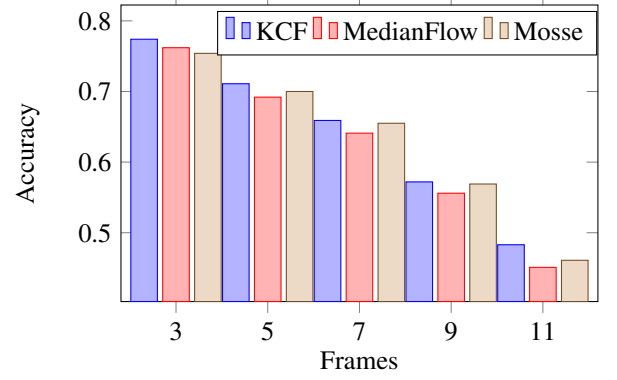


Figure 8: mAP for KCF, Median flow and Mosse tracking algorithms every n frame.

slower. This pattern was maintained in both quiet and busy environments.

The final 2D Object Detector model for the proposal was obtained after applying the auto-updated training. It also featured the Mosse tracking method with an inference of the detector every 7 frames, which provided high accuracy rates while keeping the computational cost low. The NMS threshold chosen was 0.8.

#### 4.3. Full Pipeline Experimentation

The full pipeline as an integrated system was also tested. The hardware specifications are provided in Section 3.2.

First, we used the test sequences of the UASOL dataset for benchmarking our approach. Since these sequences have reliable ground truth and the training processes did not them, they were suitable for fair testing. The sequences we used are composed of 5841 frames, which were fed to our approach in order to generate the discretized space representation for each. Two different scores were computed and reported in Table 5. First, the obstacle presence accuracy corresponds to the mean accuracy of finding an obstacle in the correct bin. The object label accuracy measures the mean accuracy of finding the correct object in the correct bin. Note, in this case, that the images were not captured by the proposed camera but we used those provided by the dataset.

In light of these quantitative results, it can be concluded that the obstacle presence detection performed with great success. In fact, 87.99% of the obstacles were properly detected and assigned to the correct bins. Furthermore, 86.97% of the obstacles were also correctly labeled.

Next, a pilot experience was carried out with a test subject. In this case, the device was set up on an individual, who walked around in the surroundings of the Alicante University campus. This environment is suitable as it features the visual appearance of an everyday outdoor environment. Random samples of this qualitative experiment are provided in Figure 9.



Table 3: Reported accuracy for each category. The first row corresponds to the model trained on the manually labeled dataset. The second row corresponds to the model trained on both manually and automatically labeled datasets.

	Bicycle	Bus	Car	Motorbike	Person	Traffic Light	Traffic Sign	mAP
2D Object Detector baseline	46.6	90.4	63.6	66.5	60.3	34.6	71.9	62.0
After Auto-updated Learning	60.8	98.1	72.2	73.0	75.9	48.4	77.2	74.2

Table 4: This table shows the FPS for different 2D Object Detector and tracker setups for high density and low density of objects in the scene.

Method	Frames/Second
2D Object Detector	17.4
Busy Environment	
KCF	18.6
MedianFlow	29.2
Mosse	29.0
Quiet Environment	
KCF	17.3
MedianFlow	19.7
Mosse	20.3

Table 5: Scores obtained by our approach in the two different test sequences of the UASOL dataset.

Sequence	Amount of frames	Obstacle Presence Accuracy (%)	Object Label Accuracy (%)
Multi-purpose I	2891	89.22	88.17
Control Tower	2950	86.77	85.77
Mean		87.99	86.97

As can be seen, the 2D Object Detector performed accurately. In every case, the objects were detected and tracked accordingly. Furthermore, some objects outside the considered range of 5 meters were detected. These objects were simply ignored. It is worth noting that there were occluded objects in the environment. Despite this being a challenging scenario, the combination of the 2D Object Detector and Mosse tracking achieved a decent performance in these cases. Nonetheless, sometimes the occluded objects were not detected at all. This is not a substantial issue as the object in the forefront is closer to the user than the occluded one, so the system does not need to notice this secondary obstacle.

The Depth Estimator also performed as expected. As shown in Figure 9, the estimated depthmaps provided poor three-dimensional representations, yet were enough for obstacle detection. Note that the accuracy of the three-dimensional representation is poorer as the distance increases. It is also worth noting that the surfaces yield huge errors and undesirable artifacts but this is not an issue as the proposed space discretization makes those errors negligible. The same conclusion could be applied to the trails pointing towards the infinite that can be observed in the outbound of some structural elements. Table 6 shows the ground truth distance to the user and the estimated

one for each sample depicted in the Figure 9 (in these frames, the actual distance to the objects were measured with a Zed Stereo camera.). Note that the estimated distances are always closer than the actual distances. This is desirable, rather than the contrary situation, in order to avoid obstacles: considering an object closer than the actual distance may not interfere adversely in the walking plan of the user. This effect arises because we deliberately reduced the distance to the objects in the training process of the depth prediction network. Nonetheless, the opposite case would be highly unlikely to lead to collisions or accidents. Due to the discretization process, minor errors in the predictions do not impact the bin assignment process. Only the object is located near the boundaries of a bin and, due to an error, is assigned to the wrong bin, could it be dangerous. However, as the user is moving through the scenario, this object is eventually going to be further from the boundaries and placed in the correct bin.

Table 6: This table shows the distances to the detected objects of the samples present in Figure 9. The actual distances were computed using the ground truth and the estimated depthmaps.

Sample	Obstacle	Actual Distance (m)	Estimated Distance (m)
1	Tree Left	4.31	4.27
	Tree Right	3.25	2.44
2	Person FG	2.28	2.09
	Person BG	2.57	N/D
	Car	3.85	3.51
3	Person FG	4.85	3.83
	Person BG	5.05	3.68
	Car	4.41	3.88
4	Tree	5.65	4.19
	Car	4.25	4.18
5	Person	1.64	1.38
6	Car	4.12	3.68

The descriptions, which are provided on demand with a simple tap on any smartwatch, were accurate enough to enable an easy understanding. For instance, the description provided by the system for the second sample of the first row in Figure 9 is: "There is an obstacle at 1.68 meters in the nearest leftmost bin, and a person at 2.09 meters in the intermediate rightmost bin, and a car at 3.51 meters in the furthest rightmost bin". The corresponding haptic feedback is a strong vibration on the left smartwatch and an intermediate one on the right one.

Note that in this case, the person in the forefront is occluding a second person and part of the car. This leads the system to ignore the second person, and provide an erroneous distance so the person and the car share part of the 2D space. Note also in this sample that the wall in the left part of the scene is referred

to as unknown object (obstacle). The walls are not considered by the 2D Object Detector despite it being an obstacle.

Table 7: Time schedule for a complete iteration of the proposed system. Note that the feedforwards of the two neural architectures are performed simultaneously, so only the heavier task is computed for the reported end-to-end time.

Process	Time (ms)
Image transference to smartphone	81.3802
Image transference to remote server	112.24
2D Object Detection feedforward	25.1051
Depthmap Estimation feedforward	55.124
Descriptions and discretized representation generation	9.5214
Transference to the smartphone	0.018
End-to-End time	258.2836

Overall, the system does not perform as expected when an object occluded another object. This makes the system perceive the farthest object as being at the same distance as the nearest object. We also noticed that, due to a depth estimation error or an object location error, some obstacles were assigned to the wrong bin. This happened when the obstacles were located near the boundaries of a bin. The flickering in the detections led it to be assigned to adjacent bins. In any event, this is a momentary issue. As the user moves, the object is no longer located in the boundaries of the bins and will be eventually placed correctly. Another minor issue is that it tends to mistake depictions of persons for actual persons. For instance in adverts and billboards. Finally, it is worth noting that, due to the discretized representation of the space, it is difficult for the user to walk through narrow gaps, for instance, between a streetlight and a wall on a narrow sidewalk.

Regarding the response time, the whole system is able to provide a response in 258.2836 milliseconds, which corresponds to 3.8717 frames per second. This includes the inference time of both networks, the description generation and the data transference overhead. The reported times comprehend the average time schedule for the depicted samples, which are shown in Table 7. Note that some processes like the 2D Object Detector and the discretized space generation and transference timings are dependent on the number of detected objects.

The user in the pilot experiment stated that the frame rate of the system was adequate to enable obstacle avoidance at normal walking speed. He also felt that the greatest limitation was his trying to interpret the feedback rather than the frame rate of the system itself. Nonetheless, he noticed an improvement in this as he became accustomed to the system.

## 5. Conclusions

In this work, a system to enhance the perception capabilities of the visually impaired is proposed. The system takes advantage of novel deep learning techniques to generate a three-dimensional representation of the scenes with semantic labels for the obstacles from the feed of a wearable camera. The labels and the estimated depth are summarized to create a simple representation of the scene, which is comprehensible and can

be quickly delivered. Two smartwatches provide haptic feedback in order to communicate the obstacles in the surroundings to the user. The training process of the Depth Estimator had to be tuned in order to make the obstacles appear slightly nearer than they actually were. The proposed 2D Object Detector performed as expected by providing localization and description of the most common objects in urban environments such as cars, buses and other people.

## 6. Supplementary Material

The video attached to this work depicts a number of sequences captured for qualitative evaluation. The video shows the detected objects bounded by green boxes. A vertical green line shows the obstacles that were detected but remained unlabeled. In the bottom left-hand corner, there is a representation of the discretized space with the obstacles superimposed. Finally, beside this, are the on-demand spoken descriptions of the environment. The main views are cycled between the color feed of the wearable camera and the estimated depth maps obtained with our model.

## 7. Future Work

We plan to carry out further deployment experimentation with actual visually impaired individuals. This would provide insights into the strengths and weaknesses of the proposed system in the context of outdoor assistance. We also plan to improve the depth map estimation and the handling of the occlusions to avoid erroneous distance measures when different objects share the same 2D space.

## Acknowledgements

This work has been supported by the Spanish Government TIN2016-76515R Grant, supported with Feder funds, the University of Alicante project GRE16-19, and by the Valencian Government project GV/2018/022. Edmanuel Cruz is funded by a Panamenian grant for PhD studies IFARHU & SENACYT 270-2016-207. This work has also been supported by a Spanish grant for PhD studies ACIF/2017/243. Thanks also to Nvidia for the generous donation of a Titan Xp and a Quadro P6000.

## References

- Bolme, D.S., Beveridge, J.R., Draper, B.A., Lui, Y.M., 2010. Visual object tracking using adaptive correlation filters. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2544–2550doi:10.1109/CVPR.2010.5539960, arXiv:1404.7584.
- Csapó, Á., Wersényi, G., Nagy, H., Stockman, T., 2015. A survey of assistive technologies and applications for blind users on mobile platforms: a review and foundation for research. *Journal on Multimodal User Interfaces* 9, 275–286. URL: <https://doi.org/10.1007/s12193-015-0182-7>, doi:10.1007/s12193-015-0182-7.
- Delahoz, Y., Labrador, M.A., 2017. A deep-learning-based floor detection system for the visually impaired, in: *2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*, pp. 883–888.

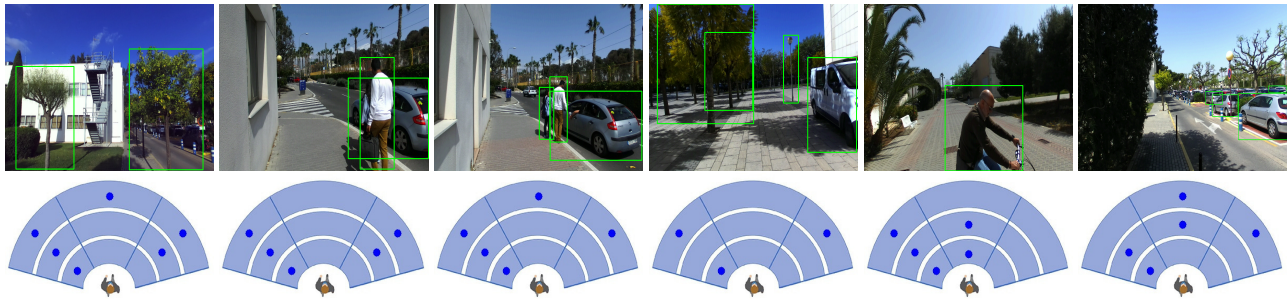


Figure 9: The first row shows sample images captured by the wearable camera and the predictions of the 2D Object Detector. The second row depicts the corresponding representation of the discretized space. Note that the detected obstacles are superimposed in blue.

Dominguez-Sanchez, A., Cazorla, M., Orts-Escolano, S., 2018. A new dataset and performance evaluation of a region-based cnn for urban object detection. *Electronics* 7. URL: <http://www.mdpi.com/2079-9292/7/11/301>, doi:10.3390/electronics7110301.

Dosovitskiy, A., Springenberg, J.T., Tatarchenko, M., Brox, T., 2014. Learning to Generate Chairs, Tables and Cars with Convolutional Networks. *ArXiv e-prints* arXiv:1411.5928.

Eigen, D., Puhrsch, C., Fergus, R., 2014. Depth map prediction from single image using a multi-scale deep network, in: *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, MIT Press, Cambridge, MA, USA. pp. 2366–2374. URL: <http://dl.acm.org/citation.cfm?id=2969033.2969091>.

Elmannai, W., Elleithy, K., 2017. Sensor-based assistive devices for visually impaired people: Current status, challenges, and future directions. *Sensors* 17. URL: <http://www.mdpi.com/1424-8220/17/3/565>, doi:10.3390/s17030565.

Hakobyan, L., Lumsden, J., O’Sullivan, D., Bartlett, H., 2013. Mobile assistive technologies for the visually impaired. *Survey of ophthalmology* 58, 513–528. doi:10.1016/j.survophtha.2012.10.004.

He, K., Zhang, X., Ren, S., Sun, J., 2015. Deep Residual Learning for Image Recognition. *ArXiv e-prints* arXiv:1512.03385.

Henriques, J.F., Caseiro, R., Martins, P., Batista, J., 2014. High-Speed Tracking with Kernelized Correlation Filters. doi:10.1109/TPAMI.2014.2345390, arXiv:1404.7584.

Jafri, R., Campos, R.L., Ali, S.A., Arabnia, H.R., 2018. Visual and infrared sensor data-based obstacle detection for the visually impaired using the google project tango tablet development kit and the unity engine. *IEEE Access* 6, 443–454.

Kalal, Z., Kalal, Z., Mikolajczyk, K., Matas, J., 2010. Forward-backward error: Automatic detection of tracking failures. IN *PROCEEDINGS OF THE 2010 20TH INTERNATIONAL CONFERENCE ON PATTERN RECOGNITION, ICPR ’10*, 2756–2759.

Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., Navab, N., 2016. Deeper depth prediction with fully convolutional residual networks. *CoRR* abs/1606.00373. URL: <http://arxiv.org/abs/1606.00373>, arXiv:1606.00373.

Lakde, C.K., Prasad, P.S., 2015. Navigation system for visually impaired people, in: *2015 International Conference on Computation of Power, Energy, Information and Communication (ICCPEIC)*, pp. 0093–0098. doi:10.1109/ICCPEIC.2015.7259447.

Lee, Y.H., Medioni, G., 2015. Wearable rgb-d indoor navigation system for the blind, in: *Agapito, L., Bronstein, M.M., Rother, C. (Eds.), Computer Vision - ECCV 2014 Workshops*, Springer International Publishing, Cham. pp. 493–508.

Lee, Y.H., Medioni, G., 2016. Rgb-d camera based wearable navigation system for the visually impaired. *Computer Vision and Image Understanding* 149, 3–20.

Lin, B.S., Lee, C.C., Chiang, P.Y., 2017. Simple smartphone-based guiding system for visually impaired people. *Sensors* 17. URL: <http://www.mdpi.com/1424-8220/17/6/1371>.

Martinez-Gonzalez, P., Oprea, S., Garcia-Garcia, A., Jover-Alvarez, A., Orts-Escolano, S., Garcia-Rodriguez, J., 2018. UnrealROX: An eXtremely Photorealistic Virtual Reality Environment for Robotics Simulations and Synthetic Data Generation. *ArXiv e-prints* arXiv:1810.06936.

Microsoft, 2018. Seeing AI. URL: <https://www.microsoft.com/en-us/seeing-ai>.

Neto, L.B., Grijalva, F., Maike, V.R.M.L., Martini, L.C., Florencio, D.,

Baranauskas, M.C.C., Rocha, A., Goldenstein, S., 2017. A kinect-based wearable face recognition system to aid visually impaired users. *IEEE Transactions on Human-Machine Systems* 47, 52–64. doi:10.1109/THMS.2016.2604367.

Poggi, M., Mattoccia, S., 2016. A wearable mobility aid for the visually impaired based on embedded 3d vision and deep learning, in: *2016 IEEE Symposium on Computers and Communication (ISCC)*, pp. 208–213. doi:10.1109/ISCC.2016.7543741.

Pradeep, V., Medioni, G., Weiland, J., 2010. Robot vision for the visually impaired, in: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops*, pp. 15–22. doi:10.1109/CVPRW.2010.5543579.

Redmon, J., Divvala, S., Girshick, R., Farhadi, A., 2016. You Only Look Once: Unified Real-Time Object Detection. *Cvpr* 2016, 779–788.

Ren, S., He, K., Girshick, R., Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks.

Saxena, A., Chung, S.H., Ng, A.Y., 2006. Learning depth from single monocular images, in: *Weiss, Y., Schölkopf, B., Platt, J.C. (Eds.), Advances in Neural Information Processing Systems 18*. MIT Press, pp. 1161–1168. URL: <http://papers.nips.cc/paper/2921-learning-depth-from-single-monocular-images.pdf>.

Tian, Y., 2014. RGB-D Sensor-Based Computer Vision Assistive Technology for Visually Impaired Persons. Springer International Publishing. pp. 173–194. URL: [https://doi.org/10.1007/978-3-319-08651-4\\_9](https://doi.org/10.1007/978-3-319-08651-4_9), doi:10.1007/978-3-319-08651-4\_9.

Wang, H., Katzschmann, R.K., Teng, S., Araki, B., Giarré, L., Rus, D., 2017. Enabling independent navigation for visually impaired people through a wearable vision-based feedback system, in: *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6533–6540. doi:10.1109/ICRA.2017.7989772.

Zwald, L., Lambert-Lacroix, S., 2012. The berhu penalty and the grouped effect.